D 8

```
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

G 3

```
**FILE**ID**LIBLEXICA






LL            IIIIII    BBBBBBBB    LL          EEEEEEEEEE  XX      XX    IIIIII      CCCCCCCC    AAAAAA
LL            IIIIII    BBBBBBBB    LL          EEEEEEEEEE  XX      XX    IIIIII      CCCCCCCC    AAAAAA
LL              II      BB      BB  LL          EE          XX      XX      II      CC          AA      AA
LL              II      BB      BB  LL          EE          XX      XX      II      CC          AA      AA
LL              II      BB      BB  LL          EE            XX  XX        II      CC          AA      AA
LL              II      BB      BB  LL          EE            XX  XX        II      CC          AA      AA
LL              II      BBBBBBBB    LL          EEEEEEEE        XX          II      CC          AA      AA
LL              II      BBBBBBBB    LL          EEEEEEEE        XX          II      CC          AA      AA
LL              II      BB      BB  LL          EE            XX  XX        II      CC          AAAAAAAAAA
LL              II      BB      BB  LL          EE            XX  XX        II      CC          AAAAAAAAAA
LL              II      BB      BB  LL          EE          XX      XX      II      CC          AA      AA      ....
LL              II      BB      BB  LL          EE          XX      XX      II      CC          AA      AA      ....
LLLLLLLLLL    IIIIII    BBBBBBBB    LLLLLLLLLL  EEEEEEEEEE  XX      XX    IIIIII      CCCCCCCC  AA      AA      ....
LLLLLLLLLL    IIIIII    BBBBBBBB    LLLLLLLLLL  EEEEEEEEEE  XX      XX    IIIIII      CCCCCCCC  AA      AA      ....


LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0 MODULE LIB$$LEXICAL ( %TITLE 'Internal routines for lexical functions'
0                 IDENT = '1-009'         ! File: LIBLEXICA.B32 Edit: STAN1009
0                 ) =
1 BEGIN
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! FACILITY:     General Utility Library, DCL
1 !
1 ! ABSTRACT:
1 !
1 !       This module contains routines which form the common kernel of
1 !       the following Run-Time Library procedures and DCL lexical functions:
1 !               LIB$GETDVI              F$GETDVI
1 !               LIB$GETJPI              F$GETJPI
1 !               LIB$GETSYI              F$GETSYI
1 !
1 !
1 ! ENVIRONMENT:  User or supervisor mode - AST reentrant
1 !
1 ! AUTHOR: Steven B. Lionel, CREATION DATE: 13-July-1982
1 !
1 ! MODIFIED BY:
1 !
1 ! 1-001 - Original.  Adapted from the DCL module LEXICON.  SBL 13-July-1982
1 ! 1-002 - Use tables in LIBGETTAB.MAR.  SBL 8-Mar-1983
1 ! 1-003 - Change string length from LNM$C_NAMLENGTH to 512.  SBL 11-Mar-1983
1 ! 1-004 - Add HEXSTR format.  SBL 20-May-1983
1 ! 1-005 - HEXSTR is now HEXSTRING.  SBL 24-May-1983
1 ! 1-006 - Add privileges TMPJNL, PRMJNL and SECURITY.  SBL 28-July-1983
1 ! 1-007 - Add new format MODE for JPI$_MODE.  Fix format HEXSTRING so that
1 !         the significant characters get returned.  SBL 9-Sep-1983
1 ! 1-008 - Add support for two new arguments to SYS$GETSYI - NODENAME and
1 !           CSIDADR.  DG 19-Oct-1983.
```

```
;   58          0058  1 | 1-009 - Fix handling of counted strings. STAN 27-Feb-1984.
;   59          0059  1 !--
;   60          0060  1
;    62        0061  1 %SBTTL 'Declarations'
;    63        0062  1 !
;    64        0063  1 ! PROLOGUE FILE:
;    65        0064  1 !
;    66        0065  1
;    67        0066  1 LIBRARY 'RTLLIB';                                       ! SYS$LIBRARY:LIB.L32
;    68        0067  1 REQUIRE 'RTLIN:LIBPROLOG';                              ! LIB$ definitions
;    69        0138  1
;    70        0139  1 !
;    71        0140  1 ! LINKAGES:
;    72        0141  1 !
;    73        0142  1
;    74        0143  1 LINKAGE
;    75        0144  1     CALL_LEXICAL = CALL;
;    76        0145  1
;    77        0146  1 !
;    78        0147  1 ! TABLE OF CONTENTS:
;    79        0148  1 !
;    80        0149  1
;    81        0150  1 FORWARD ROUTINE
;    82        0151  1     LIB$$GETDVI: CALL_LEXICAL,                          ! Get Device Information
;    83        0152  1     LIB$$GETJPI: CALL_LEXICAL,                          ! Get Job/Process Information
;    84        0153  1     LIB$$GETSYI: CALL_LEXICAL,                          ! Get System Imformation
;    85        0154  1     LIB$$FORMAT_RESULT: NOVALUE;                        ! Format result
;    86        0155  1
;    87        0156  1 !
;    88        0157  1 ! MACROS:
;    89        0158  1 !
;    90        0159  1 !       NONE
;    91        0160  1 !
;    92        0161  1 ! EQUATED SYMBOLS:
;    93        0162  1 !
;    94        0163  1 !       NONE
;    95        0164  1 !
;    96        0165  1 ! FIELDS:
;    97        0166  1 !
;    98        0167  1 !       NONE
;    99        0168  1 !
;   100        0169  1 ! OWN STORAGE:
;   101        0170  1 !
;   102        0171  1 !       NONE
;   103        0172  1 !
;   104        0173  1 ! EXTERNAL REFERENCES:
;   105        0174  1 !
;   106        0175  1
;   107        0176  1 EXTERNAL ROUTINE
;   108        0177  1     OTS$CVT_L_TZ;                 ! Convert to hex format
;   109        0178  1
;   110        0179  1 EXTERNAL
;   111        0180  1     LIB$$AB_GETDVI_TABLE,         ! Table of $GETDVI codes and types
;   112        0181  1     LIB$$AB_GETJPI_TABLE,         ! Table of $GETJPI codes and types
;   113        0182  1     LIB$$AB_GETSYI_TABLE;         ! Table of $GETSYI codes and types
```

```
1 %SBTTL 'LIB$$GETDVI - Internal routine for LIB$GETDVI'
1 GLOBAL ROUTINE LIB$$GETDVI (
1       ITEM_CODE: WORD SIGNED,               ! $GETSYI Item code
1       RET_STRING: REF VECTOR [, BYTE],      ! Return string buffer
1       RET_NUMBER: REF BLOCK [, BYTE],       ! Return numeric buffer
1       RET_LENGTH: REF VECTOR [, WORD],      ! Returned length
1       RET_TYPE: REF VECTOR [, LONG],        ! Returned type code
1       EVENT_FLAG,                           ! Event flag to use
1       CHANNEL: WORD,                        ! Channel number
1       DEVNAM_DESCR: REF BLOCK [, BYTE]      ! Device name descriptor
1       ): CALL_LEXICAL =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Kernel routine called from LIB$GETDVI and DCL to get device and
1 !       volume information.  See LIB$GETDVI for more information.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret-status.wlc.v = LIB$$GETDVI (
1 !                               item-code.rw.v,
1 !                               ret-string.wt.r,
1 !                               ret-number.wq.r,
1 !                               ret-length.wwu.r,
1 !                               ret-type.wl.r,
1 !                               event-flag.rl.v,
1 !                               channel.rwu.v,
1 !                               devnam-descr.rt.ds)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       item-code               The $GETDVI item code
1 !
1 !       ret-string              A string of length 512 into which
1 !                               is placed the string-formatted value.
1 !
1 !       ret-number              A quadword into which is placed the numeric
1 !                               value, if any.
1 !
1 !       ret-length              A word into which is placed the length of
1 !                               the string in ret-string.
1 !
1 !       ret-type                A longword into which is placed the type
1 !                               code for the value being returned.  The
1 !                               codes are LIB$K_FMT_xxx values defined
1 !                               in LIBFMTDEF.SDL.
1 !
1 !       event-flag              A longword event flag number to use for
1 !                               the $GETDVI.
1 !
1 !       channel                 A word containing the channel to inquire
1 !                               about.
1 !
1 !       devnam-descr            A string descriptor for the device name
1 !                               being inquired about.
1 !
```

```
1 ! IMPLICIT I PUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       SS$_xxx         Any error status from $GETDVIW
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
```

```
;      0258  2      BEGIN
;      0259  2
;      0260  2      !+
;      0261  2      ! Declare fieldset that defines the layout of a GETDVI_ITEM
;      0262  2      !-
;      0263  2
;      0264  2      FIELD
;      0265  2          GETDVI_ITEM_FIELDSET =
;      0266  2              SET
;      0267  2              W_ITEM = [0,0,16,1],          ! DVI$_ item code value
;      0268  2              B_TYPE = [0,16,8,0],          ! LIB$K_FMT_ type code
;      0269  2              A_NEXT = [3,0,0,0]            ! Offset of next item
;      0270  2              TES;
;      0271  2
```

```
LOCAL
    TABLE_ENTRY: REF BLOCK [, BYTE] FIELD (GETDVI_ITEM_FIELDSET),
                                        ! Current table entry
    DUMMY_ENTRY: BLOCK [3, BYTE] FIELD (GETDVI_ITEM_FIELDSET),
    ITEM_LIST: BLOCK [16, BYTE],        ! Item list for $GETDVI
    IOSB: VECTOR [4, WORD],             ! Status block
    RET_STATUS;                         ! Return status

!+
! Look up ITEM_CODE in LIB$$AB_GETDVI_TABLE.
!-

TABLE_ENTRY = LIB$$AB_GETDVI_TABLE;           ! Get first element.

WHILE .TABLE_ENTRY [W_ITEM] NEQ .ITEM_CODE
DO
    BEGIN
    TABLE_ENTRY = TABLE_ENTRY [A_NEXT];       ! Get next item
    IF .TABLE_ENTRY [W_ITEM] EQL 0            ! No more items?
    THEN
        BEGIN
        TABLE_ENTRY = DUMMY_ENTRY;            ! Use dummy table entry
        DUMMY_ENTRY [B_TYPE] = LIB$K_FMT_BINARY;
        EXITLOOP;
        END;
    END;

!+
! Store type code.
!-

RET_TYPE [0] = .TABLE_ENTRY [B_TYPE];

!+
! Fill in ITEM_LIST and do the $GETDVI.
!-

ITEM_LIST [0,16,16,0] = .ITEM_CODE;           ! Item code
IF .TABLE_ENTRY [B_TYPE] LEQ LIB$K_FMT_MAXSTRING    ! Is it a string?
THEN
    BEGIN
    ITEM_LIST [4,0,32,0] = RET_STRING [0];    ! Return buffer
    ITEM_LIST [0,0,16,0] = 512;               ! Buffer size
    IF .TABLE_ENTRY [B_TYPE] EQL LIB$K_FMT_HEXSTRING
    THEN
        ITEM_LIST [0,0,16,0] = 256;           ! Can't cvt more than 256 bytes
    END
ELSE
    BEGIN
    RET_NUMBER [0,0,32,0] = 0;                ! Zero the buffer
    RET_NUMBER [4,0,32,0] = 0;
    ITEM_LIST [4,0,32,0] = RET_NUMBER [0,0,0,0];    ! Return buffer
    ITEM_LIST [0,0,16,0] = 8;                 ! Buffer size (Quadword)
    END;
ITEM_LIST [8,0,32,0] = RET_LENGTH [0];        ! Return length
ITEM_LIST [12,0,32,0] = 0;                    ! End of list
```

```
;   263         P 0329  2       RET_STATUS = $GETDVIW (EFN = .EVENT_FLAG, CHAN = .CHANNEL,
;   264           0330  2           DEVNAM = DEVNAM_DESCR [0,0,0,0], ITMLST = ITEM_LIST, IOSB = IOSB);
;   265           0331  2
;   266           0332  2       IF .RET_STATUS
;   267           0333  2       THEN
;   268           0334  2           RET_STATUS = .IOSB [0];
;   269           0335  2
;   270           0336  2       !+
;   271           0337  2       ! Check for errors.
;   272           0338  2       !-
;   273           0339  2
;   274           0340  2       IF NOT .RET_STATUS
;   275           0341  2       THEN
;   276           0342  2           RETURN .RET_STATUS;                 ! Return with error code
;   277           0343  2
;   278           0344  2       !+
;   279           0345  2       ! Now call LIB$$FORMAT_RESULT to format the result, if necessary.
;   280           0346  2       !-
;   281           0347  2
;   282           0348  2       LIB$$FORMAT_RESULT (RET_STRING [0], RET_NUMBER [0,0,0,0], RET_LENGTH [0],
;   283           0349  2           RET_TYPE [0]);
;   284           0350  2
;   285           0351  2       RETURN SS$_NORMAL;
;   286           0352  2
;   287           0353  1       END;                                            ! End of routine LIB$$GETDVI


                                                                        .TITLE  LIB$$LEXICAL Internal routines for lexical func
;                                                                               tions
                                                                        .IDENT  \1-009\

                                                                        .EXTRN  OTS$CVT_L_TZ, LIB$$AB_GETDVI_TABLE
                                                                        .EXTRN  LIB$$AB_GETJPI_TABLE
                                                                        .EXTRN  LIB$$AB_GETSYI_TABLE
                                                                        .EXTRN  SYS$GETDVIW

                                                                        .PSECT  _LIB$CODE,NOWRT,  SHR,  PIC,2

                                            0000 00000                  .ENTRY  LIB$$GETDVI, Save nothing                       ; 0184
                       5E               1C   C2 00002                   SUBL2   #28, SP                                         ;
                       50 00000000G     00   9E 00005                   MOVAB   LIB$$AB_GETDVI_TABLE, TABLE_ENTRY               ; 0284
                 04    AC               60   B1 0000C 1$:               CMPW    (TABLE_ENTRY), ITEM_CODE                        ; 0286
                                        0E   13 00010                   BEQL    2$                                              ;
                       50               03   C0 00012                   ADDL2   #3, TABLE_ENTRY                                 ; 0289
                                        60   B5 00015                   TSTW    (TABLE_ENTRY)                                   ; 0290
                                        F3   12 00017                   BNEQ    1$                                              ;
                       50               6E   9E 00019                   MOVAB   DUMMY_ENTRY, TABLE_ENTRY                        ; 0293
                 02    AE               08   90 0001C                   MOVB    #8, DUMMY_ENTRY+2                               ; 0294
                 14    BC         02    A0   9A 00020 2$:               MOVZBL  2(TABLE_ENTRY), @RET_TYPE                       ; 0303
                 0E    AE         04    AC   B0 00025                   MOVW    ITEM_CODE, ITEM_LIST+2                          ; 0309
                       03         02    A0   91 0002A                   CMPB    2(TABLE_ENTRY), #3                              ; 0310
                                        19   1A 0002E                   BGTRU   3$                                              ;
                 10    AE         08    AC   D0 00030                   MOVL    RET_STRING, ITEM_LIST+4                         ; 0313
                 0C    AE       0200    8F   B0 00035                   MOVW    #512, ITEM_LIST                                 ; 0314
                       02         02    A0   91 0003B                   CMPB    2(TABLE_ENTRY), #2                              ; 0315
                                        16   12 0003F                   BNEQ    4$                                              ;
                 0C    AE       0100    8F   B0 00041                   MOVW    #256, ITEM_LIST                                 ; 0317
```

```
                                   0E  11 00047          BRB      4$                                      ; 0310
                     50      0C    AC  D0 00049 3$:      MOVL     RET_NUMBER, R0                          ; 0321
                                   60  7C 0004D          CLRQ     (R0)                                    ;
                10   AE            50  D0 0004F          MOVL     R0, ITEM_LIST+4                         ; 0323
                0C   AE            08  B0 00053          MOVW     #8, ITEM_LIST                           ; 0324
                14   AE      10    AC  D0 00057 4$:      MOVL     RET_LENGTH, ITEM_LIST+8                 ; 0326
                             18    AE  D4 0005C          CLRL     ITEM_LIST+12                            ; 0327
                                   7E  7C 0005F          CLRQ     -(SP)                                   ; 0330
                                   7E  D4 00061          CLRL     -(SP)                                   ;
                             10    AE  9F 00063          PUSHAB   IOSB                                    ;
                             1C    AE  9F 00066          PUSHAB   ITEM_LIST                               ;
                             20    AC  DD 00069          PUSHL    DEVNAM_DESCR                            ;
                     7E      1C    AC  3C 0006C          MOVZWL   CHANNEL, -(SP)                          ;
                             18    AC  DD 00070          PUSHL    EVENT_FLAG                              ;
        00000000G    00            08  FB 00073          CALLS    #8, SYS$GETDVIW                         ;
                     17            50  E9 0007A          BLBC     RET_STATUS, 5$                          ; 0332
                     50      04    AE  3C 0007D          MOVZWL   IOSB, RET_STATUS                        ; 0334
                     10            50  E9 00081          BLBC     RET_STATUS, 5$                          ; 0340
                     7E      10    AC  7D 00084          MOVQ     RET_LENGTH, -(SP)                       ; 0349
                     7E      08    AC  7D 00088          MOVQ     RET_STRING, -(SP)                       ;
            0000V    CF            04  FB 0008C          CALLS    #4, LIB$$FORMAT_RESULT                  ;
                     50            01  D0 00091          MOVL     #1, R0                                  ; 0351
                                   04     00094 5$:      RET                                              ; 0353

; Routine Size:  149 bytes,    Routine Base:  _LIB$CODE + 0000
```

```
%SBTTL 'LIB$$GETJPI - Internal routine for LIB$GETJPI'
GLOBAL ROUTINE LIB$$GETJPI (
    ITEM_CODE: WORD SIGNED,                 ! $GETJPI Item code
    RET_STRING: REF VECTOR [, BYTE],        ! Return string buffer
    RET_NUMBER: REF BLOCK [, BYTE],         ! Return numeric buffer
    RET_LENGTH: REF VECTOR [, WORD],        ! Returned length
    RET_TYPE: REF VECTOR [, LONG],          ! Returned type code
    EVENT_FLAG,                             ! Event flag to use
    PIDADDR,                                ! Address of PID
    PRCNAM_DESCR: REF BLOCK [, BYTE]        ! Process name descriptor
    ): CALL_LEXICAL =

!++
! FUNCTIONAL DESCRIPTION:
!
!       Kernel routine called from LIB$GETJPI and DCL to get job and
!       process information.  See LIB$GETJPI for more information.
!
! CALLING SEQUENCE:
!
!       ret-status.wlc.v = LIB$$GETJPI (
!                                       item-code.rw.v,
!                                       ret-string.wt.r,
!                                       ret-number.wq.r,
!                                       ret-length.wwu.r,
!                                       ret-type.wl.r,
!                                       event-flag.rl.v,
!                                       pidaddr.ra.v,
!                                       prcnam-descr.rt.ds)
!
! FORMAL PARAMETERS:
!
!       item-code               The $GETJPI item code
!
!       ret-string              A string of length 512 into which
!                               is placed the string-formatted value.
!
!       ret-number              A quadword into which is placed the numeric
!                               value, if any.
!
!       ret-length              A word into which is placed the length of
!                               the string in ret-string.
!
!       ret-type                A longword into which is placed the type
!                               code for the value being returned.  The
!                               codes are LIB$K_FMT_xxx values defined
!                               in LIBFMTDEF.SDL.
!
!       event-flag              A longword event flag number to use for
!                               the $GETJPI.
!
!       pidaddr                 The address of the PID, if any, being inquired
!                               about.
!
!       prcnam-descr            A string descriptor for the process name
!                               being inquired about, if any
!
```

```
;  346          0411   1 ! IMPLICIT INPUTS:
;  347          0412   1 !
;  348          0413   1 !       NONE
;  349          0414   1 !
;  350          0415   1 ! IMPLICIT OUTPUTS:
;  351          0416   1 !
;  352          0417   1 !       NONE
;  353          0418   1 !
;  354          0419   1 ! COMPLETION STATUS:
;  355          0420   1 !
;  356          0421   1 !       SS$_NORMAL       Normal successful completion
;  357          0422   1 !       SS$_xxx          Any error status from $GETJPIW
;  358          0423   1 !
;  359          0424   1 ! SIDE EFFECTS:
;  360          0425   1 !
;  361          0426   1 !       NONE
;  362          0427   1 !
;  363          0428   1 !--
;  365         0429  2      BEGIN
;  366         0430  2
;  367         0431  2      !+
;  368         0432  2      ! Declare fieldset that defines the layout of a GETJPI_ITEM.
;  369         0433  2      !-
;  370         0434  2
;  371         0435  2      FIELD
;  372         0436  2          GETJPI_ITEM_FIELDSET =
;  373         0437  2              SET
;  374         0438  2              W_ITEM = [0,0,16,1],        ! JPI$_ item code value
;  375         0439  2              B_TYPE = [0,16,8,0],        ! LIB$K_FMT_ type code
;  376         0440  2              A_NEXT = [3,0,0,0]          ! Offset of next item
;  377         0441  2              TES;
;  378         0442  2
```

```
2       LOCAL
2           TABLE_ENTRY: REF BLOCK [, BYTE] FIELD (GETJPI_ITEM_FIELDSET),
2                                           ! Current table entry
2           DUMMY_ENTRY: BLOCK [3, BYTE] FIELD (GETJPI_ITEM_FIELDSET),
2           ITEM_LIST: BLOCK [16, BYTE],    ! Item list for $GETJPI
2           IOSB: VECTOR [4, WORD],         ! Status block
2           RET_STATUS;                     ! Return status
2
2       !+
2       ! Look up ITEM_CODE in LIB$$AB_GETJPI_TABLE.
2       !-
2
2       TABLE_ENTRY = LIB$$AB_GETJPI_TABLE;         ! Get first element.
2
2       WHILE .TABLE_ENTRY [W_ITEM] NEQ .ITEM_CODE
2       DO
3           BEGIN
3           TABLE_ENTRY = TABLE_ENTRY [A_NEXT];     ! Get next item
3           IF .TABLE_ENTRY [W_ITEM] EQL 0          ! No more items?
3           THEN
4               BEGIN
4               TABLE_ENTRY = DUMMY_ENTRY;  ! Use dummy entry
4               TABLE_ENTRY [B_TYPE] = LIB$K_FMT_BINARY;
4               EXITLOOP;
3               END;
2           END;
2
2       !+
2       ! Store type code.
2       !-
2
2       RET_TYPE [0] = .TABLE_ENTRY [B_TYPE];
2
2       !+
2       ! Fill in ITEM_LIST and do the $GETJPI.
2       !-
2
2       ITEM_LIST [0,16,16,0] = .ITEM_CODE; ! Item code
2       IF .TABLE_ENTRY [B_TYPE] LEQ LIB$K_FMT_MAXSTRING
2       THEN
3           BEGIN
3           ITEM_LIST [4,0,32,0] = RET_STRING [0];  ! Return buffer
3           ITEM_LIST [0,0,16,0] = 512;             ! Buffer size
3           IF .TABLE_ENTRY [B_TYPE] EQL LIB$K_FMT_HEXSTRING
3           THEN
3               ITEM_LIST [0,0,16,0] = 256;         ! Can't cvt more than 256 bytes
3           END
2       ELSE
3           BEGIN
3           RET_NUMBER [0,0,32,0] = 0;              ! Zero the buffer
3           RET_NUMBER [4,0,32,0] = 0;
3           ITEM_LIST [4,0,32,0] = RET_NUMBER [0,0,0,0];     ! Return buffer
3           ITEM_LIST [0,0,16,0] = 8;               ! Buffer size (Quadword)
2           END;
2       ITEM_LIST [8,0,32,0] = RET_LENGTH [0];      ! Return length
2       ITEM_LIST [12,0,32,0] = 0;                  ! End of list
```

```
;  437        P 0500   2     RET_STATUS = $GETJPIW (EFN = .EVENT_FLAG, PIDADR = .PIDADDR,
;  438          0501   2         PRCNAM = PRCNAM_DESCR [0,0,0,0], ITMLST = ITEM_LIST, IOSB = IOSB);
;  439          0502   2
;  440          0503   2     IF .RET_STATUS
;  441          0504   2     THEN
;  442          0505   2         RET_STATUS = .IOSB [0];
;  443          0506   2
;  444          0507   2     !+
;  445          0508   2     ! Check for errors.
;  446          0509   2     !-
;  447          0510   2
;  448          0511   2     IF NOT .RET_STATUS
;  449          0512   2     THEN
;  450          0513   2         RETURN .RET_STATUS;
;  451          0514   2
;  452          0515   2     !+
;  453          0516   2     ! Now call LIB$$FORMAT_RESULT to format the result, if necessary.
;  454          0517   2     !-
;  455          0518   2
;  456          0519   2     LIB$$FORMAT_RESULT (RET_STRING [0], RET_NUMBER [0,0,0,0], RET_LENGTH [0],
;  457          0520   2         RET_TYPE [0]);
;  458          0521   2
;  459          0522   2     RETURN SS$_NORMAL;
;  460          0523   2
;  461          0524   1     END;                                        ! End of routine LIB$$GETJPI


                                                            .EXTRN  SYS$GETJPIW

                                       0000 00000           .ENTRY  LIB$$GETJPI, Save nothing                       ; 0355
            5E                     1C    C2 00002           SUBL2   #28, SP                                           ;
            50 00000000G           00    9E 00005           MOVAB   LIB$$AB_GETJPI_TABLE, TABLE_ENTRY                 ; 0455
      04    AC                     60    B1 0000C 1$:       CMPW    (TABLE_ENTRY), ITEM_CODE                          ; 0457
                                   0E    13 00010           BEQL    2$                                                ;
            50                     03    C0 00012           ADDL2   #3, TABLE_ENTRY                                   ; 0460
                                   60    B5 00015           TSTW    (TABLE_ENTRY)                                     ; 0461
                                   F3    12 00017           BNEQ    1$                                                ;
            50                     6E    9E 00019           MOVAB   DUMMY_ENTRY, TABLE_ENTRY                          ; 0464
      02    A0                     08    90 0001C           MOVB    #8, 2(TABLE_ENTRY)                                ; 0465
      14    BC          02         A0    9A 00020 2$:       MOVZBL  2(TABLE_ENTRY), @RET_TYPE                         ; 0474
      0E    AE          04         AC    B0 00025           MOVW    ITEM_CODE, ITEM_LIST+2                            ; 0480
            03          02         A0    91 0002A           CMPB    2(TABLE_ENTRY), #3                                ; 0481
                                   19    1A 0002E           BGTRU   3$                                                ;
      10    AE          08         AC    D0 00030           MOVL    RET_STRING, ITEM_LIST+4                           ; 0484
      0C    AE        0200         8F    B0 00035           MOVW    #512, ITEM_LIST                                   ; 0485
            02          02         A0    91 0003B           CMPB    2(TABLE_ENTRY), #2                                ; 0486
                                   16    12 0003F           BNEQ    4$                                                ;
      0C    AE        0100         8F    B0 00041           MOVW    #256, ITEM_LIST                                   ; 0488
                                   0E    11 00047           BRB     4$                                                ; 0481
            50          0C         AC    D0 00049 3$:       MOVL    RET_NUMBER, R0                                    ; 0492
                                   60    7C 0004D           CLRQ    (R0)                                              ;
      10    AE                     50    D0 0004F           MOVL    R0, ITEM_LIST+4                                   ; 0494
      0C    AE                     08    B0 00053           MOVW    #8, ITEM_LIST                                     ; 0495
      14    AE          10         AC    D0 00057 4$:       MOVL    RET_LENGTH, ITEM_LIST+8                           ; 0497
                        18         AE    D4 0005C           CLRL    ITEM_LIST+12                                      ; 0498
                                   7E    7C 0005F           CLRQ    -(SP)                                             ; 0501
```

```
                                    0C    AE  9F 00061          PUSHAB  IOSB                                        ;
                                    18    AE  9F 00064          PUSHAB  ITEM_LIST                                   ;
                         7E         1C    AC  7D 00067          MOVQ    PIDADDR, -(SP)                              ;
                                    18    AC  DD 0006B          PUSHL   EVENT_FLAG                                  ;
              00000000G  00               07  FB 0006E          CALLS   #7, SYS$GETJPIW                             ;
                         17               50  E9 00075          BLBC    RET_STATUS, 5$                              ; 0503
                         50         04    AE  3C 00078          MOVZWL  IOSB, RET_STATUS                            ; 0505
                         10               50  E9 0007C          BLBC    RET_STATUS, 5$                              ; 0511
                         7E         10    AC  7D 0007F          MOVQ    RET_LENGTH, -(SP)                           ; 0520
                         7E         08    AC  7D 00083          MOVQ    RET_STRING, -(SP)                           ;
                  0000V  CF               04  FB 00087          CALLS   #4, LIB$$FORMAT_RESULT                      ;
                         50               01  D0 0008C          MOVL    #1, R0                                      ; 0522
                                              04 0008F  5$:     RET                                                 ; 0524

; Routine Size:  144 bytes,    Routine Base:  _LIB$CODE + 0095
```

```
%SBTTL 'LIB$$GETSYI - Internal routine for LIB$GETSYI'
GLOBAL ROUTINE LIB$$GETSYI (
    ITEM_CODE: WORD,                       ! $GETSYI Item code
    RET_STRING: REF VECTOR [, BYTE],       ! Return string buffer
    RET_NUMBER: REF VECTOR [, BYTE],       ! Return numeric buffer
    RET_LENGTH: REF VECTOR [, WORD],       ! Returned length
    RET_TYPE: REF VECTOR [, LONG],         ! Returned type code
    EVENT_FLAG,                            ! Event flag to use
    CSIDADR,                               ! Address of CSID src/dest
    NODENAME_DESCR: REF BLOCK [, BYTE]     ! Nodedname descriptor
    ): CALL_LEXICAL =

!++
! FUNCTIONAL DESCRIPTION:
!
!       Kernel routine called from LIB$GETSYI and DCL to get system-wide
!       information.  See LIB$GETSYI for more information.
!
! CALLING SEQUENCE:
!
!       ret-status.wlc.v = LIB$$GETSYI (
!                               item-code.rw.v,
!                               ret-string.wt.r,
!                               ret-number.wq.r,
!                               ret-length.wwu.r,
!                               ret-type.wl.r,
!                               event-flag.rl.v,
!                               csidadr.ra.v,
!                               nodename-descr.rt.ds)
!
! FORMAL PARAMETERS:
!
!       item-code               The $GETSYI item code
!
!       ret-string              A string of length 512 into which
!                               is placed the string-formatted value.
!
!       ret-number              A quadword into which is placed the numeric
!                               value, if any.
!
!       ret-length              A word into which is placed the length of
!                               the string in ret-string.
!
!       ret-type                A longword into which is placed the type
!                               code for the value being returned.  The
!                               codes are LIB$K_FMT_xxx values defined
!                               in LIBFMTDEF.SDL.
!
!       event-flag              A longword event flag number to use for
!                               the $GETSYI.
!
!       csidadr                 The address of CSID source/destination,
!                               if any, being inquired about.
!
!       nodename-descr          A string descriptor for the nodename being
!                               inquired about, in any.
!
```

```
 ; 520          0582  1 ! IMPLICIT INPUTS:
 ; 521          0583  1 !
 ; 522          0584  1 !       NONE
 ; 523          0585  1 !
 ; 524          0586  1 ! IMPLICIT OUTPUTS:
 ; 525          0587  1 !
 ; 526          0588  1 !       NONE
 ; 527          0589  1 !
 ; 528          0590  1 ! COMPLETION STATUS:
 ; 529          0591  1 !
 ; 530          0592  1 !       SS$_NORMAL      Normal successful completion
 ; 531          0593  1 !       SS$_xxx         Any error status from $GETSYIW
 ; 532          0594  1 !
 ; 533          0595  1 ! SIDE EFFECTS:
 ; 534          0596  1 !
 ; 535          0597  1 !       NONE
 ; 536          0598  1 !
 ; 537          0599  1 !--
```

```
;   539            0600  2       BEGIN
;   540            0601  2
;   541            0602  2       !+
;   542            0603  2       ! Declare fieldset that defines the layout of a GETSYI_ITEM.
;   543            0604  2       !-
;   544            0605  2
;   545            0606  2       FIELD
;   546            0607  2           GETSYI_ITEM_FIELDSET =
;   547            0608  2               SET
;   548            0609  2               W_ITEM = [0,0,16,0],          ! SYI$_ item code
;   549            0610  2               B_TYPE = [0,16,8,0],          ! LIB$K_FMT_ type code
;   550            0611  2               A_NEXT = [3,0,0,0]            ! Offset of next item
;   551            0612  2               TES;
;   552            0613  2
```

```
  2        LOCAL
  2            TABLE_ENTRY: REF BLOCK [, BYTE] FIELD (GETSYI_ITEM_FIELDSET),
  2                                            ! Current table entry
  2            DUMMY_ENTRY: BLOCK [3, BYTE] FIELD (GETSYI_ITEM_FIELDSET),
  2            IOSB: VECTOR [4, WORD],         ! I/O status block
  2            ITEM_LIST: BLOCK [16, BYTE],    ! Item list for $GETSYI
  2            RET_STATUS;                     ! Return status
  2
  2        !+
  2        ! Look up ITEM_CODE in LIB$$AB_GETSYI_TABLE.
  2        !-
  2
  2        TABLE_ENTRY = LIB$$AB_GETSYI_TABLE;          ! Get first element.
  2
  2        WHILE .TABLE_ENTRY [W_ITEM] NEQ .ITEM_CODE
  2        DO
  3            BEGIN
  3            TABLE_ENTRY = TABLE_ENTRY [A_NEXT];     ! Get next item
  3            IF .TABLE_ENTRY [W_ITEM] EQL 0          ! No more items?
  3            THEN
  4                BEGIN
  4                TABLE_ENTRY = DUMMY_ENTRY;          ! Use dummy entry
  4                TABLE_ENTRY [B_TYPE] = LIB$K_FMT_BINARY;
  4                EXITLOOP;
  3                END;
  2            END;
  2
  2        !+
  2        ! Store type code.
  2        !-
  2
  2        RET_TYPE [0] = .TABLE_ENTRY [B_TYPE];
  2
  2        !+
  2        ! Fill in ITEM_LIST and do the $GETSYI.
  2        !-
  2
  2        ITEM_LIST [0,16,16,0] = .ITEM_CODE; ! Item code
  2        IF .TABLE_ENTRY [B_TYPE] LEQ LIB$K_FMT_MAXSTRING
  2        THEN
  3            BEGIN
  3            ITEM_LIST [4,0,32,0] = RET_STRING [0];  ! Return buffer
  3            ITEM_LIST [0,0,16,0] = 512;             ! Buffer size
  3            IF .TABLE_ENTRY [B_TYPE] EQL LIB$K_FMT_HEXSTRING
  3            THEN
  3                ITEM_LIST [0,0,16,0] = 256;         ! Can't cvt more than 256 bytes
  3            END
  2        ELSE
  3            BEGIN
  3            ITEM_LIST [4,0,32,0] = RET_NUMBER [0];  ! Return buffer
  3            ITEM_LIST [0,0,16,0] = 8;               ! Buffer size (Quadword)
  2            END;
  2        ITEM_LIST [8,0,32,0] = RET_LENGTH [0];      ! Return length
  2        ITEM_LIST [12,0,32,0] = 0;                  ! End of list
  2
P 2        RET_STATUS = $GETSYIW (EFN = .EVENT_FLAG, CSIDADR = .CSIDADR,
  2            NODENAME = .NODENAME_DESCR,ITMLST = ITEM_LIST, IOSB = IOSB);
```

```
;  611          0671  2
;  612          0672  2       IF .RET_STATUS
;  613          0673  2       THEN
;  614          0674  2           RET_STATUS = .IOSB [0];
;  615          0675  2
;  616          0676  2       IF NOT .RET_STATUS
;  617          0677  2       THEN
;  618          0678  2           RETURN .RET_STATUS;
;  619          0679  2
;  620          0680  2       !+
;  621          0681  2       ! Now call LIBSSFORMAT_RESULT to format the result.
;  622          0682  2       !-
;  623          0683  2
;  624          0684  2       LIBSSFORMAT_RESULT (RET_STRING [0], RET_NUMBER [0], RET_LENGTH [0],
;  625          0685  2           RET_TYPE [0]);
;  626          0686  2
;  627          0687  2       RETURN SS$_NORMAL;
;  628          0688  2
;  629          0689  1       END;                                           ! End of routine LIBSSGETSYI


                                                                     .EXTRN  SYSSGETSYIW

                                               0000 00000            .ENTRY  LIBSSGETSYI, Save nothing                          ; 0526
                    5E                    1C    C2 00002             SUBL2   #28, SP                                            ;
                    50 00000000G          00    9E 00005             MOVAB   LIBSSAB_GETSYI_TABLE, TABLE_ENTRY                  ; 0626
               04   AC                    60    B1 0000C 1$:         CMPW    (TABLE_ENTRY), ITEM_CODE                           ; 0628
                                          0E    13 00010             BEQL    2$                                                 ;
                    50                    03    C0 00012             ADDL2   #3, TABLE_ENTRY                                    ; 0631
                                          60    B5 00015             TSTW    (TABLE_ENTRY)                                      ; 0632
                                          F3    12 00017             BNEQ    1$                                                 ;
                    50                    6E    9E 00019             MOVAB   DUMMY_ENTRY, TABLE_ENTRY                           ; 0635
               02   A0                    08    90 0001C             MOVB    #8, 2(TABLE_ENTRY)                                 ; 0636
               14   BC             02     A0    9A 00020 2$:         MOVZBL  2(TABLE_ENTRY), @RET_TYPE                          ; 0645
               06   AE             04     AC    B0 00025             MOVW    ITEM_CODE, ITEM_LIST+2                             ; 0651
                    03             02     A0    91 0002A             CMPB    2(TABLE_ENTRY), #3                                 ; 0652
                                          19    1A 0002E             BGTRU   3$                                                 ;
               08   AE             08     AC    D0 00030             MOVL    RET_STRING, ITEM_LIST+4                            ; 0655
               04   AE           0200     8F    B0 00035             MOVW    #512, ITEM_LIST                                    ; 0656
                    02             02     A0    91 0003B             CMPB    2(TABLE_ENTRY), #2                                 ; 0657
                                          11    12 0003F             BNEQ    4$                                                 ;
               04   AE           0100     8F    B0 00041             MOVW    #256, ITEM_LIST                                    ; 0659
                                          09    11 00047             BRB     4$                                                 ; 0652
               08   AE             0C     AC    D0 00049 3$:         MOVL    RET_NUMBER, ITEM_LIST+4                            ; 0663
               04   AE                    08    B0 0004E             MOVW    #8, ITEM_LIST                                      ; 0664
               0C   AE             10     AC    D0 00052 4$:         MOVL    RET_LENGTH, ITEM_LIST+8                            ; 0666
                                   10     AE    D4 00057             CLRL    ITEM_LIST+12                                       ; 0667
                                          7E    7C 0005A             CLRQ    -(SP)                                              ; 0670
                                   1C     AE    9F 0005C             PUSHAB  IOSB                                               ;
                                   10     AE    9F 0005F             PUSHAB  ITEM_LIST                                          ;
                    7E             1C     AC    7D 00062             MOVQ    CSIDADR, -(SP)                                     ;
                                   18     AC    DD 00066             PUSHL   EVENT_FLAG                                         ;
        00000000G   00                    07    FB 00069             CALLS   #7, SYSSGETSYIW                                    ;
                    17                    50    E9 00070             BLBC    RET_STATUS, 5$                                     ; 0672
                    50             14     AE    3C 00073             MOVZWL  IOSB, RET_STATUS                                   ; 0674
                    10                    50    E9 00077             BLBC    RET_STATUS, 5$                                     ; 0676
```

```
                              7E        10      AC  7D 0007A          MOVQ     RET_LENGTH, -(SP)                       ; 0685
                              7E        08      AC  7D 0007E          MOVQ     RET_STRING, -(SP)                       ;
                      0000V   CF                04  FB 00082          CALLS    #4, LIB$$FORMAT_RESULT                  ;
                              50                01  D0 00087          MOVL     #1, R0                                  ; 0687
                                                    04 0008A 5$:      RET                                              ; 0689

; Routine Size:  139 bytes,    Routine Base:  _LIB$CODE + 0125
```

```
1 %SBTTL 'LIB$$FORMAT_RESULT - Format the result'
1 GLOBAL ROUTINE LIB$$FORMAT_RESULT (
1     RET_STRING: REF VECTOR [, BYTE],      ! Return string buffer
1     RET_NUMBER: REF BLOCK [, BYTE],       ! Return numeric buffer
1     RET_LENGTH: REF VECTOR [, WORD],      ! Returned length
1     RET_TYPE: REF VECTOR [, LONG]         ! Returned type code
1     ): NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Called by LIB$$GETxxI routines to convert the value returned
1 !       by the $GETxxI service to the appropriate string format.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       CALL LIB$$FORMAT_RESULT (ret-string.mt.r, ret-number.rq.r,
1 !               ret-length.mwu.r, ret-type.rl.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       ret-string      A string of length 512 into which
1 !                       is placed the formatted result.  If the
1 !                       value type is already a string, the value
1 !                       is in ret-string.
1 !
1 !       ret-number      A quadword containing the numeric value to
1 !                       be formatted.
1 !
1 !       ret-length      A word containing the current length of the
1 !                       string in ret-string, if any, and into which
1 !                       is stored the length of the formatted result.
1 !
1 !       ret-type        A longword indicating the type of the value.
1 !                       The type codes are LIB$K_FMT_xxx symbols and
1 !                       are defined in LIBFMTDEF.SDL.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
```

```
LOCAL
    CTRSTR_DESCR: BLOCK [8, BYTE],  ! FAO control string descriptor
    OUTSTR_DESCR: BLOCK [8, BYTE],  ! Output string descriptor
    PRMLST: VECTOR [4, LONG];       ! FAOL parameter list

!+
! Table of ACP type names.
!-

BIND
    ACP_TYPES = UPLIT BYTE (
        %ASCIC'UNKNOWN',      ! 0
        %ASCIC'F11V1',0,0,  $ASSUME (DVI$C_ACP_F11V1, EQL, 1)
        %ASCIC'F11V2',0,0,  $ASSUME (DVI$C_ACP_F11V2, EQL, 2)
        %ASCIC'MTA',0,0,0,0,$ASSUME (DVI$C_ACP_MTA, EQL, 3)
        %ASCIC'NET',0,0,0,0,$ASSUME (DVI$C_ACP_NET, EQL, 4)
        %ASCIC'REM',0,0,0,0,$ASSUME (DVI$C_ACP_REM, EQL, 5)
        %ASCIC'JNL',0,0,0,0)$ASSUME (DVI$C_ACP_JNL, EQL, 6)
        : VECTOR [, LONG];

!+
! Table of process state names.
!-

BIND
    STATES = UPLIT BYTE (
        %ASCIC'UNKNOWN',      ! 0
        %ASCIC'COLPG',0,0,  $ASSUME (SCH$C_COLPG, EQL, 1)
        %ASCIC'MWAIT',0,0,  $ASSUME (SCH$C_MWAIT, EQL, 2)
        %ASCIC'CEF',0,0,0,0,$ASSUME (SCH$C_CEF, EQL, 3)
        %ASCIC'PFW',0,0,0,0,$ASSUME (SCH$C_PFW, EQL, 4)
        %ASCIC'LEF',0,0,0,0,$ASSUME (SCH$C_LEF, EQL, 5)
        %ASCIC'LEFO',0,0,0, $ASSUME (SCH$C_LEFO, EQL, 6)
        %ASCIC'HIB',0,0,0,0,$ASSUME (SCH$C_HIB, EQL, 7)
        %ASCIC'HIBO',0,0,0, $ASSUME (SCH$C_HIBO, EQL, 8)
        %ASCIC'SUSP',0,0,0, $ASSUME (SCH$C_SUSP, EQL, 9)
        %ASCIC'SUSPO',0,0,  $ASSUME (SCH$C_SUSPO, EQL, 10)
        %ASCIC'FPG',0,0,0,0,$ASSUME (SCH$C_FPG, EQL, 11)
        %ASCIC'COM',0,0,0,0,$ASSUME (SCH$C_COM, EQL, 12)
        %ASCIC'COMO',0,0,0, $ASSUME (SCH$C_COMO, EQL, 13)
        %ASCIC'CUR',0,0,0,0)$ASSUME (SCH$C_CUR, EQL, 14)
        : VECTOR [, LONG];

!+
! Table of process mode names.
!-

BIND
    MODES = UPLIT BYTE (
        %ASCIC'OTHER',        $ASSUME (JPI$K_OTHER, EQL, 0)
        %ASCIC'NETWORK',      $ASSUME (JPI$K_NETWORK, EQL, 1)
        %ASCIC'BATCH',        $ASSUME (JPI$K_BATCH, EQL, 2)
        %ASCIC'INTERACTIVE',$ASSUME (JPI$K_INTERACTIVE, EQL, 3)
        0)              ! End of list
        : VECTOR [, BYTE];

!+
! Table of privilege names.
!-

GLOBAL BIND
    LIB$$AT_PRV_NAMES = UPLIT BYTE (
        %ASCIC'CMKRNL',$ASSUME ($BITPOSITION(PRV$V_CMKRNL), EQL, 0)
        %ASCIC'CMEXEC',$ASSUME ($BITPOSITION(PRV$V_CMEXEC), EQL, 1)
        %ASCIC'SYSNAM',$ASSUME ($BITPOSITION(PRV$V_SYSNAM), EQL, 2)
        %ASCIC'GRPNAM',$ASSUME ($BITPOSITION(PRV$V_GRPNAM), EQL, 3)
        %ASCIC'ALLSPOOL',$ASSUME ($BITPOSITION(PRV$V_ALLSPOOL), EQL, 4)
        %ASCIC'DETACH',$ASSUME ($BITPOSITION(PRV$V_DETACH), EQL, 5)
        %ASCIC'DIAGNOSE',$ASSUME ($BITPOSITION(PRV$V_DIAGNOSE), EQL, 6)
        %ASCIC'LOG_IO',$ASSUME ($BITPOSITION(PRV$V_LOG_IO), EQL, 7)
        %ASCIC'GROUP',$ASSUME ($BITPOSITION(PRV$V_GROUP), EQL, 8)
        %ASCIC'NOACNT',$ASSUME ($BITPOSITION(PRV$V_NOACNT), EQL, 9)
        %ASCIC'PRMCEB',$ASSUME ($BITPOSITION(PRV$V_PRMCEB), EQL, 10)
        %ASCIC'PRMMBX',$ASSUME ($BITPOSITION(PRV$V_PRMMBX), EQL, 11)
        %ASCIC'PSWAPM',$ASSUME ($BITPOSITION(PRV$V_PSWAPM), EQL, 12)
        %ASCIC'SETPRI',$ASSUME ($BITPOSITION(PRV$V_SETPRI), EQL, 13)
        %ASCIC'SETPRV',$ASSUME ($BITPOSITION(PRV$V_SETPRV), EQL, 14)
        %ASCIC'TMPMBX',$ASSUME ($BITPOSITION(PRV$V_TMPMBX), EQL, 15)
        %ASCIC'WORLD',$ASSUME ($BITPOSITION(PRV$V_WORLD), EQL, 16)
        %ASCIC'MOUNT',$ASSUME ($BITPOSITION(PRV$V_MOUNT), EQL, 17)
        %ASCIC'OPER',$ASSUME ($BITPOSITION(PRV$V_OPER), EQL, 18)
        %ASCIC'EXQUOTA',$ASSUME ($BITPOSITION(PRV$V_EXQUOTA), EQL, 19)
        %ASCIC'NETMBX',$ASSUME ($BITPOSITION(PRV$V_NETMBX), EQL, 20)
        %ASCIC'VOLPRO',$ASSUME ($BITPOSITION(PRV$V_VOLPRO), EQL, 21)
        %ASCIC'PHY_IO',$ASSUME ($BITPOSITION(PRV$V_PHY_IO), EQL, 22)
        %ASCIC'BUGCHK',$ASSUME ($BITPOSITION(PRV$V_BUGCHK), EQL, 23)
        %ASCIC'PRMGBL',$ASSUME ($BITPOSITION(PRV$V_PRMGBL), EQL, 24)
        %ASCIC'SYSGBL',$ASSUME ($BITPOSITION(PRV$V_SYSGBL), EQL, 25)
        %ASCIC'PFNMAP',$ASSUME ($BITPOSITION(PRV$V_PFNMAP), EQL, 26)
        %ASCIC'SHMEM',$ASSUME ($BITPOSITION(PRV$V_SHMEM), EQL, 27)
        %ASCIC'SYSPRV',$ASSUME ($BITPOSITION(PRV$V_SYSPRV), EQL, 28)
        %ASCIC'BYPASS',$ASSUME ($BITPOSITION(PRV$V_BYPASS), EQL, 29)
        %ASCIC'SYSLCK',$ASSUME ($BITPOSITION(PRV$V_SYSLCK), EQL, 30)
        %ASCIC'SHARE',$ASSUME ($BITPOSITION(PRV$V_SHARE), EQL, 31)
        %ASCIC'UPGRADE',$ASSUME ($BITPOSITION(PRV$V_UPGRADE), EQL, 0) ! 32
        %ASCIC'DOWNGRADE',$ASSUME ($BITPOSITION(PRV$V_DOWNGRADE), EQL, 1) ! 33
        %ASCIC'GRPPRV',$ASSUME ($BITPOSITION(PRV$V_GRPPRV), EQL, 2) ! 34
        %ASCIC'READALL',$ASSUME ($BITPOSITION(PRV$V_READALL), EQL, 3) ! 35
        %ASCIC'TMPJNL',$ASSUME ($BITPOSITION(PRV$V_TMPJNL), EQL, 4) ! 36
        %ASCIC'PRMJNL',$ASSUME ($BITPOSITION(PRV$V_PRMJNL), EQL, 5) ! 37
        %ASCIC'SECURITY',$ASSUME ($BITPOSITION(PRV$V_SECURITY), EQL, 6) ! 38
        0)      ! End of list
        : VECTOR [, BYTE];

!+
! Fill in constant descriptor information.
!-

CTRSTR_DESCR [DSC$B_DTYPE] = DSC$K_DTYPE_T;
CTRSTR_DESCR [DSC$B_CLASS] = DSC$K_CLASS_S;
OUTSTR_DESCR [DSC$B_DTYPE] = DSC$K_DTYPE_T;
OUTSTR_DESCR [DSC$B_CLASS] = DSC$K_CLASS_S;
OUTSTR_DESCR [DSC$W_LENGTH] = 512;
OUTSTR_DESCR [DSC$A_POINTER] = RET_STRING [0];
```

```
;  802        0861  2
;  803        0862  2          !+
;  804        0863  2          ! Select the formatting action appropriate to the item type.
;  805        0864  2          !-
;  806        0865  2
;  807        0866  2          CASE .RET_TYPE [0] FROM LIB$K_FMT_MIN TO LIB$K_FMT_MAX OF
;  808        0867  2              SET
;  809        0868  2
;  810        0869  2              [LIB$K_FMT_BINARY]:
;  811        0870  3                  BEGIN
;  812        0871  3                  CTRSTR_DESCR [DSC$W_LENGTH] = %CHARCOUNT ('!UL');
;  813        0872  3                  CTRSTR_DESCR [DSC$A_POINTER] = UPLIT BYTE ('!UL');
;  814        0873  3                  PRMLST [0] = .RET_NUMBER [0,0,32,0];
;  815      P 0874  3                  $FAOL (CTRSTR = CTRSTR_DESCR [0,0,0,0],
;  816      P 0875  3                          OUTLEN = RET_LENGTH [0],
;  817      P 0876  3                          OUTBUF = OUTSTR_DESCR [0,0,0,0],
;  818        0877  3                          PRMLST = PRMLST [0]);
;  819        0878  2                  END;
;  820        0879  2
;  821        0880  2              [LIB$K_FMT_BOOLEAN]:
;  822        0881  3                  BEGIN
;  823        0882  3                  IF .RET_NUMBER [0,0,1,0]
;  824        0883  3                  THEN
;  825        0884  4                      BEGIN
;  826        0885  4                      RET_LENGTH [0] = %CHARCOUNT ('TRUE');
;  827        0886  4                      CH$MOVE (%CHARCOUNT ('TRUE'), UPLIT BYTE ('TRUE'),
;  828        0887  4                          RET_STRING [0]);
;  829        0888  4                      END
;  830        0889  3                  ELSE
;  831        0890  4                      BEGIN
;  832        0891  4                      RET_LENGTH [0] = %CHARCOUNT ('FALSE');
;  833        0892  4                      CH$MOVE (%CHARCOUNT ('FALSE'), UPLIT BYTE ('FALSE'),
;  834        0893  4                          RET_STRING [0]);
;  835        0894  3                      END;
;  836        0895  2                  END;
;  837        0896  2
;  838        0897  2              [LIB$K_FMT_HEX]:
;  839        0898  3                  BEGIN
;  840        0899  3                  CTRSTR_DESCR [DSC$W_LENGTH] = %CHARCOUNT ('!XL');
;  841        0900  3                  CTRSTR_DESCR [DSC$A_POINTER] = UPLIT BYTE ('!XL');
;  842        0901  3                  PRMLST [0] = .RET_NUMBER [0,0,32,0];
;  843      P 0902  3                  $FAOL (CTRSTR = CTRSTR_DESCR [0,0,0,0],
;  844      P 0903  3                          OUTLEN = RET_LENGTH [0],
;  845      P 0904  3                          OUTBUF = OUTSTR_DESCR [0,0,0,0],
;  846        0905  3                          PRMLST = PRMLST [0]);
;  847        0906  2                  END;
;  848        0907  2
;  849        0908  2              [LIB$K_FMT_HEXSTRING]:
;  850        0909  3                  BEGIN
;  851        0910  3                  LOCAL
;  852        0911  3                      TEMP_STRING: VECTOR [512, BYTE];
;  853        0912  3
;  854        0913  3                  CTRSTR_DESCR [DSC$W_LENGTH] = .RET_LENGTH [0] * 2;
;  855        0914  3                  CTRSTR_DESCR [DSC$A_POINTER] = TEMP_STRING;
;  856        0915  3                  OTS$CVT_L_TZ (RET_STRING [0], CTRSTR_DESCR [0,0,0,0],
;  857        0916  3                      .CTRSTR_DESCR [DSC$W_LENGTH], .RET_LENGTH [0]);
;  858        0917  3                  RET_LENGTH [0] = .RET_LENGTH [0] * 2;
```

```
:   3              CH$MOVE (.CTRSTR_DESCR [DSC$W_LENGTH], TEMP_STRING,
:   3                  RET_STRING [0]);
:   2              END;
:   2
:   2          [LIB$K_FMT_DATE]:
:   3              BEGIN
:   3              CTRSTR_DESCR [DSC$W_LENGTH] = %CHARCOUNT ('!%D');
:   3              CTRSTR_DESCR [DSC$A_POINTER] = UPLIT BYTE ('!%D');
:   3              PRMLST [0] = RET_NUMBER [0,0,0,0];
: P 3              $FAOL (CTRSTR = CTRSTR_DESCR [0,0,0,0],
: P 3                      OUTLEN = RET_LENGTH [0],
: P 3                      OUTBUF = OUTSTR_DESCR [0,0,0,0],
:   3                      PRMLST = PRMLST [0]);
:   2              END;
:   2
:   2          [LIB$K_FMT_PRIVILEGE]:
:   3              BEGIN
:   3              LOCAL
:   3                  STRING_PTR,                 ! Pointer to current char in string
:   3                  PRV_NAME: REF VECTOR [, BYTE], ! Privilege name
:   3                  PRV;                        ! Current privilege number
:   3
:   3              STRING_PTR = RET_STRING [0]; ! First position in string
:   3              PRV_NAME = LIB$$AT_PRV_NAMES [0];    ! First privilege name
:   3              INCRU PRV FROM 0 TO 63 DO
:   4                  BEGIN
:   4                  IF .PRV_NAME [0] EQL 0  ! No more defined privilege names
:   4                  THEN
:   4                      EXITLOOP;
:   4                  IF .RET_NUMBER [0,.PRV,1,0]
:   4                  THEN
:   5                      BEGIN
:   5                      STRING_PTR = CH$MOVE (.PRV_NAME [0], PRV_NAME [1],
:   5                          .STRING_PTR);
:   5                      CH$WCHAR_A (%C',', STRING_PTR);
:   4                      END;
:   4                  PRV_NAME = .PRV_NAME + .PRV_NAME [0] + 1;       ! Next name
:   3                  END;
:   3              IF .STRING_PTR NEQA RET_STRING [0]
:   3              THEN
:   3                  STRING_PTR = .STRING_PTR - 1;   ! Trim trailing comma
:   3              RET_LENGTH [0] = .STRING_PTR - RET_STRING [0];      ! Get length
:   2              END;
:   2
:   2          [LIB$K_FMT_UIC]:
:   3              BEGIN
:   3              CTRSTR_DESCR [DSC$W_LENGTH] = %CHARCOUNT ('!%U');
:   3              CTRSTR_DESCR [DSC$A_POINTER] = UPLIT BYTE ('!%U');
:   3              PRMLST [0] = .RET_NUMBER [0,0,32,0];
: P 3              $FAOL (CTRSTR = CTRSTR_DESCR [0,0,0,0],
: P 3                      OUTLEN = RET_LENGTH [0],
: P 3                      OUTBUF = OUTSTR_DESCR [0,0,0,0],
:   3                      PRMLST = PRMLST [0]);
:   2              END;
:   2
:   2          [LIB$K_FMT_PROT, LIB$K_FMT_VPROT]:
:   3              BEGIN
```

```
		LOCAL
		    PSTRING: VECTOR [24, BYTE],
		    PSTRING_PTR,
		    PROT_CHARS: REF VECTOR [, BYTE],
		    PROT_FIELD: BLOCK [1, BYTE];

		!+
		! Select the correct protection codes for files or volumes.
		!-

		IF .RET_TYPE EQL LIB$K_FMT_PROT
		THEN
		    PROT_CHARS = UPLIT BYTE ('RWED')
		ELSE
		    PROT_CHARS = UPLIT BYTE ('RWLP');

		PSTRING_PTR = PSTRING [0];
		INCR I FROM 0 TO 3 BY 1 DO
		    BEGIN
		    LOCAL
			THIS_STRING: REF VECTOR [, BYTE];
		    PRMLST [.I] = .PSTRING_PTR;
		    THIS_STRING = .PSTRING_PTR;
		    CH$WCHAR_A (0, PSTRING_PTR);		! Set initial length
		    PROT_FIELD = .RET_NUMBER [0,.I*4,4,0] XOR %X'F';
		    IF .PROT_FIELD NEQ 0
		    THEN
			BEGIN
			CH$WCHAR_A (%C'=', PSTRING_PTR);
			THIS_STRING [0] = .THIS_STRING [0] + 1;
			INCR J FROM 0 TO 3 BY 1 DO
			    BEGIN
			    IF .PROT_FIELD [0,.J,1,0]
			    THEN
				BEGIN
				CH$WCHAR_A (.PROT_CHARS [.J], PSTRING_PTR);
				THIS_STRING [0] = .THIS_STRING [0] + 1;
				END;
			    END;
			END;
		    END;
		CTRSTR_DESCR [DSC$W_LENGTH] =
		    %CHARCOUNT ('SYSTEM!AC,OWNER!AC,GROUP!AC,WORLD!AC');
		CTRSTR_DESCR [DSC$A_POINTER] =
		    UPLIT BYTE ('SYSTEM!AC,OWNER!AC,GROUP!AC,WORLD!AC');
		$FAOL (CTRSTR = CTRSTR_DESCR [0,0,0,0],
			OUTLEN = RET_LENGTH [0],
			OUTBUF = OUTSTR_DESCR [0,0,0,0],
			PRMLST = PRMLST [0]);
		END;


	    [LIB$K_FMT_ACP]:
		BEGIN
		LOCAL
		    ACPTYP_PTR: REF VECTOR [, BYTE];
		IF .RET_NUMBER [0,0,32,0] GTRU DVI$C_ACP_JNL
```

```
                THEN
                    ACPTYP_PTR = ACP_TYPES [0]      ! Illegal
                ELSE
                    ACPTYP_PTR = ACP_TYPES [2*.RET_NUMBER [0,0,32,0]];
                RET_LENGTH [0] = .ACPTYP_PTR [0];
                CH$MOVE (.RET_LENGTH [0], ACPTYP_PTR [1], RET_STRING [0]);
                END;

            [LIB$K_FMT_STATE]:
                BEGIN
                LOCAL
                    STATE_PTR: REF VECTOR [, BYTE];
                IF .RET_NUMBER [0,0,32,0] GTRU SCH$C_CUR
                THEN
                    STATE_PTR = STATES [0]  ! Illegal
                ELSE
                    STATE_PTR = STATES [2*.RET_NUMBER [0,0,32,0]];
                RET_LENGTH [0] = .STATE_PTR [0];
                CH$MOVE (.RET_LENGTH [0], STATE_PTR [1], RET_STRING [0]);
                END;

            [LIB$K_FMT_MODE]:
                BEGIN
                LOCAL
                    MODE_PTR: REF VECTOR [, BYTE];
                MODE_PTR = MODES [0];
                INCRU I FROM 1 TO .RET_NUMBER [0,0,32,0] DO
                    BEGIN
                    IF .MODE_PTR [0] EQLU 0
                    THEN
                        BEGIN
                        MODE_PTR = MODES [0];       ! Invalid, use OTHER
                        EXITLOOP;
                        END;
                    MODE_PTR = .MODE_PTR + .MODE_PTR [0] + 1; ! Skip this string
                    END;
                RET_LENGTH [0] = .MODE_PTR [0];
                CH$MOVE (.RET_LENGTH [0], MODE_PTR [1], RET_STRING [0]);
                END;

            [LIB$K_FMT_PSTRING,LIB$K_FMT_ASCIC]:    ! Strip trailing blanks
                BEGIN
                DECRU I FROM .RET_LENGTH [0] TO 1 DO
                    IF .RET_STRING [.I] EQL %C' '
                    THEN
                        RET_LENGTH [0] = .RET_LENGTH [0] - 1;
                END;

!  Note: $GETJPI does not return counted strings.  It returns
!        these strings as zero-padded strings.
!
!           [LIB$K_FMT_ASCIC]:
!               BEGIN
!               RET_LENGTH [0] = .RET_STRING [0];
!               CH$MOVE (.RET_LENGTH [0], RET_STRING [1], RET_STRING [0]);
!               END;

```

```
                        2            [INRANGE, OUTRANGE]:
                        2                ;
                        2
                        2            TES;
                        2
                        2        RETURN;
                        2
                        1        END;                                          ! End of routine LIB$$FORMAT_RESULT;



                    4E  57  4F  4E  4B  4E  55  07  001B0 P.AAA:  .ASCII  <7>\UNKNOWN\
                            31  56  31  31  46  05  001B8         .ASCII  <5>\F11V1\
                                            00  00  001BE         .BYTE   0, 0
                            32  56  31  31  46  05  001C0         .ASCII  <5>\F11V2\
                                            00  00  001C6         .BYTE   0, 0
                                    41  54  4D  03  001C8         .ASCII  <3>\MTA\
                                    00  00  00  00  001CC         .BYTE   0, 0, 0, 0
                                    54  45  4E  03  001D0         .ASCII  <3>\NET\
                                    00  00  00  00  001D4         .BYTE   0, 0, 0, 0
                                    4D  45  52  03  001D8         .ASCII  <3>\REM\
                                    00  00  00  00  001DC         .BYTE   0, 0, 0, 0
                                    4C  4E  4A  03  001E0         .ASCII  <3>\JNL\
                                    00  00  00  00  001E4         .BYTE   0, 0, 0, 0
                    4E  57  4F  4E  4B  4E  55  07  001E8 P.AAB:  .ASCII  <7>\UNKNOWN\
                            47  50  4C  4F  43  05  001F0         .ASCII  <5>\COLPG\
                                            00  00  001F6         .BYTE   0, 0
                            54  49  41  57  4D  05  001F8         .ASCII  <5>\MWAIT\
                                            00  00  001FE         .BYTE   0, 0
                                    46  45  43  03  00200         .ASCII  <3>\CEF\
                                    00  00  00  00  00204         .BYTE   0, 0, 0, 0
                                    57  46  50  03  00208         .ASCII  <3>\PFW\
                                    00  00  00  00  0020C         .BYTE   0, 0, 0, 0
                                    46  45  4C  03  00210         .ASCII  <3>\LEF\
                                    00  00  00  00  00214         .BYTE   0, 0, 0, 0
                                4F  46  45  4C  04  00218         .ASCII  <4>\LEFO\
                                        00  00  00  0021D         .BYTE   0, 0, 0
                                    42  49  48  03  00220         .ASCII  <3>\HIB\
                                    00  00  00  00  00224         .BYTE   0, 0, 0, 0
                                4F  42  49  48  04  00228         .ASCII  <4>\HIBO\
                                        00  00  00  0022D         .BYTE   0, 0, 0
                                50  53  55  53  04  00230         .ASCII  <4>\SUSP\
                                        00  00  00  00235         .BYTE   0, 0, 0
                            4F  50  53  55  53  05  00238         .ASCII  <5>\SUSPO\
                                            00  00  0023E         .BYTE   0, 0
                                    47  50  46  03  00240         .ASCII  <3>\FPG\
                                    00  00  00  00  00244         .BYTE   0, 0, 0, 0
                                    4D  4F  43  03  00248         .ASCII  <3>\COM\
                                    00  00  00  00  0024C         .BYTE   0, 0, 0, 0
                                4F  4D  4F  43  04  00250         .ASCII  <4>\COMO\
                                        00  00  00  00255         .BYTE   0, 0, 0
                                    52  55  43  03  00258         .ASCII  <3>\CUR\
                                    00  00  00  00  0025C         .BYTE   0, 0, 0, 0
                            52  45  48  54  4F  05  00260 P.AAC:  .ASCII  <5>\OTHER\
                    4B  52  4F  57  54  45  4E  07  00266         .ASCII  <7>\NETWORK\
                            48  43  54  41  42  05  0026E         .ASCII  <5>\BATCH\
    45  56  49  54  43  41  52  45  54  4E  49  0B  00274         .ASCII  <11>\INTERACTIVE\
                                                        00  00280         .BYTE   0
                                4C  4E  52  4B  4D  43  06  00281 P.AAD:  .ASCII  <6>\CMKRNL\
                                43  45  58  45  4D  43  06  00288         .ASCII  <6>\CMEXEC\
                                4D  41  4E  53  59  53  06  0028F         .ASCII  <6>\SYSNAM\
                                4D  41  4E  50  52  47  06  00296         .ASCII  <6>\GRPNAM\
                        4C  4F  4F  50  53  4C  4C  41  08  0029D         .ASCII  <8>\ALLSPOOL\
                                48  43  41  54  45  44  06  002A6         .ASCII  <6>\DETACH\
                        45  53  4F  4E  47  41  49  44  08  002AD         .ASCII  <8>\DIAGNOSE\
                                4F  49  5F  47  4F  4C  06  002B6         .ASCII  <6>\LOG_IO\
                                    50  55  4F  52  47  05  002BD         .ASCII  <5>\GROUP\
                                54  4E  43  41  4F  4E  06  002C3         .ASCII  <6>\NOACNT\
                                42  45  43  4D  52  50  06  002CA         .ASCII  <6>\PRMCEB\
                                58  42  4D  4D  52  50  06  002D1         .ASCII  <6>\PRMMBX\
                                4D  50  41  57  53  50  06  002D8         .ASCII  <6>\PSWAPM\
                                49  52  50  54  45  53  06  002DF         .ASCII  <6>\SETPRI\
                                56  52  50  54  45  53  06  002E6         .ASCII  <6>\SETPRV\
                                58  42  4D  50  4D  54  06  002ED         .ASCII  <6>\TMPMBX\
                                    44  4C  52  4F  57  05  002F4         .ASCII  <5>\WORLD\
                                    54  4E  55  4F  4D  05  002FA         .ASCII  <5>\MOUNT\
                                        52  45  50  4F  04  00300         .ASCII  <4>\OPER\
                            41  54  4F  55  51  58  45  07  00305         .ASCII  <7>\EXQUOTA\
                                58  42  4D  54  45  4E  06  0030D         .ASCII  <6>\NETMBX\
                                4F  52  50  4C  4F  56  06  00314         .ASCII  <6>\VOLPRO\
                                4F  49  5F  59  48  50  06  0031B         .ASCII  <6>\PHY_IO\
                                4B  48  43  47  55  42  06  00322         .ASCII  <6>\BUGCHK\
                                4C  42  47  4D  52  50  06  00329         .ASCII  <6>\PRMGBL\
                                4C  42  47  53  59  53  06  00330         .ASCII  <6>\SYSGBL\
                                50  41  4D  4E  46  50  06  00337         .ASCII  <6>\PFNMAP\
                                    4D  45  4D  48  53  05  0033E         .ASCII  <5>\SHMEM\
                                56  52  50  53  59  53  06  00344         .ASCII  <6>\SYSPRV\
                                53  53  41  50  59  42  06  0034B         .ASCII  <6>\BYPASS\
                                4B  43  4C  53  59  53  06  00352         .ASCII  <6>\SYSLCK\
                                    45  52  41  48  53  05  00359         .ASCII  <5>\SHARE\
                            45  44  41  52  47  50  55  07  0035F         .ASCII  <7>\UPGRADE\
                    45  44  41  52  47  4E  57  4F  44  09  00367         .ASCII  <9>\DOWNGRADE\
                                56  52  50  50  52  47  06  00371         .ASCII  <6>\GRPPRV\
                            4C  4C  41  44  41  45  52  07  00378         .ASCII  <7>\READALL\
                                4C  4E  4A  50  4D  54  06  00380         .ASCII  <6>\TMPJNL\
                                4C  4E  4A  4D  52  50  06  00387         .ASCII  <6>\PRMJNL\
                        59  54  49  52  55  43  45  53  08  0038E         .ASCII  <8>\SECURITY\
                                                        00  00397         .BYTE   0
                                                4C  55  21  00398 P.AAE:  .ASCII  \!UL\
                                            45  55  52  54  0039B P.AAF:  .ASCII  \TRUE\
                                        45  53  4C  41  46  0039F P.AAG:  .ASCII  \FALSE\
                                                4C  58  21  003A4 P.AAH:  .ASCII  \!XL\
                                                44  25  21  003A7 P.AAI:  .ASCII  \!%D\
                                                55  25  21  003AA P.AAJ:  .ASCII  \!%U\
                                            44  45  57  52  003AD P.AAK:  .ASCII  \RWED\
                                            50  4C  57  52  003B1 P.AAL:  .ASCII  \RWLP\
52  45  4E  57  4F  2C  43  41  21  4D  45  54  53  59  53  003B5 P.AAM:  .ASCII  \SYSTEM!AC,OWNER!AC,GROUP!AC,WORLD!AC\
4F  57  2C  43  41  21  50  55  4F  52  47  2C  43  41  21  003C4
                                    43  41  21  44  4C  52  003D3

                                                                  ACP_TYPES=          P.AAA
                                                                  STATES=             P.AAB
                                                                  MODES=              P.AAC
                                                                  LIB$$AT_PRV_NAMES== P.AAD
```

```
                                                                   L  5
LIB$LEXICAL      Internal routines for lexical functions         16-Sep-1984 01:04:32    VAX-11 Bliss-32 V4.0-742              Page  31
1-009            LIB$FORMAT_RESULT - Format the result           14-Sep-1984 12:39:06    [LIBRTL.SRC]LIBLEXICA.B32;1                (12)

                                                                          .EXTRN  SYS$FAOL

                                                   07FC 00000             .ENTRY  LIB$FORMAT_RESULT, Save R2,R3,R4,R5,R6,R7,-; 0691
                                                                                  R8,R9,R10                                :
                              5A      FDD1    CF    9E 00002              MOVAB   ACP_TYPES, R10                           :
                              5E      FDE0    CE    9E 00007              MOVAB   -54$(SP), SP                             :
                        FA    AD      010E    8F    B0 0000C              MOVW    #270, CTRSTR_DESCR+2                     : 0855
                        F0    AD  010E0200    8F    D0 00012              MOVL    #17695232, OUTSTR_DESCR                  : 0859
                              56        04    AC    D0 0001A              MOVL    RET_STRING, R6                           : 0860
                        F4    AD              56    D0 0001E              MOVL    R6, OUTSTR_DESCR+4                       :
               12             00        10    BC    CF 00022              CASEL   @RET_TYPE, #0, #18                       : 0866
    01A9       005D         01A9            01BC       00027 1$:          .WORD   38$-1$,-                                 :
    01BC       01BC         01BC            01BC       0002F                      34$-1$,-                                 :
    0085       0033         004C            0027       00037                      7$-1$,-                                  :
    00D9       00D9         00CC            0097       0003F                      34$-1$,-                                 :
               0179         0158            0142       00047                      38$-1$,-                                 :
                                                                                  38$-1$,-                                 :
                                                                                  38$-1$,-                                 :
                                                                                  38$-1$,-                                 :
                                                                                  38$-1$,-                                 :
                                                                                  2$-1$,-                                  :
                                                                                  5$-1$,-                                  :
                                                                                  3$-1$,-                                  :
                                                                                  8$-1$,-                                  :
                                                                                  10$-1$,-                                 :
                                                                                  15$-1$,-                                 :
                                                                                  16$-1$,-                                 :
                                                                                  16$-1$,-                                 :
                                                                                  24$-1$,-                                 :
                                                                                  26$-1$,-                                 :
                                                                                  29$-1$                                   :
                                                       04 0004D           RET                                              :
                        F8    AD              03    B0 0004E 2$:          MOVW    #3, CTRSTR_DESCR                         : 0871
                        FC    AD      01E8    CA    9E 00052              MOVAB   P.AAE, CTRSTR_DESCR+4                    : 0872
                                              23    11 00058              BRB     6$                                       : 0873
                              0A        08    BC    E9 0005A 3$:          BLBC    @RET_NUMBER, 4$                          : 0882
                        0C    BC              04    B0 0005E              MOVW    #4, @RET_LENGTH                          : 0885
                              66      01EB    CA    D0 00062              MOVL    P.AAF, (R6)                              : 0887
                                                    04 00067              RET                                              : 0882
                        0C    BC              05    B0 00068 4$:          MOVW    #5, @RET_LENGTH                          : 0891
               66     01EF    CA              05    28 0006C              MOVC3   #5, P.AAG, (R6)                          : 0893
                                                    04 00072              RET                                              : 0866
                        F8    AD              03    B0 00073 5$:          MOVW    #3, CTRSTR_DESCR                         : 0899
                        FC    AD      01F4    CA    9E 00077              MOVAB   P.AAH, CTRSTR_DESCR+4                    : 0900
                        E0    AD        08    BC    D0 0007D 6$:          MOVL    @RET_NUMBER, PRMLST                      : 0901
                                              37    11 00082              BRB     9$                                       : 0905
          F8   AD       0C    BC              02    A5 00084 7$:          MULW3   #2, @RET_LENGTH, CTRSTR_DESCR            : 0913
                        FC    AD              6E    9E 0008A              MOVAB   TEMP_STRING, CTRSTR_DESCR+4              : 0914
                              7E        0C    BC    3C 0008E              MOVZWL  @RET_LENGTH, -(SP)                       : 0916
                              7E        F8    AD    3C 00092              MOVZWL  CTRSTR_DESCR, -(SP)                      :
                                        F8    AD    9F 00096              PUSHAB  CTRSTR_DESCR                             : 0915
                                              56    DD 00099              PUSHL   R6                                       :
               00000000G      00              04    FB 0009B              CALLS   #4, OTS$CVT_L_TZ                         :
                        0C    BC              02    A4 000A2              MULW2   #2, @RET_LENGTH                          : 0917
               66             6E        F8    AD    28 000A6              MOVC3   CTRSTR_DESCR, TEMP_STRING, (R6)          : 0919
                                                    04 000AB              RET                                              : 0866
                        F8    AD              03    B0 000AC 8$:          MOVW    #3, CTRSTR_DESCR                         : 0924
                        FC    AD      01F7    CA    9E 000B0              MOVAB   P.AAI, CTRSTR_DESCR+4                    : 0925
```

```
                    E0   AD       08   AC  D0 000B6        MOVL    RET_NUMBER, PRMLST                           ; 0926
                                     0097  31 000BB  9$:   BRW     23$                                          ; 0930
                         53            56  D0 000BE 10$:   MOVL    R6, STRING_PTR                               ; 0940
                         57     00D1   CA  9E 000C1        MOVAB   LIB$$AT_PRV_NAMES, PRV_NAME                  ; 0941
                                       59  D4 000C6        CLRL    PRV                                          ; 0947
                         58            67  9A 000C8 11$:   MOVZBL  (PRV_NAME), R8                               ; 0944
                                       19  13 000CB        BEQL    13$
               08   08   BC            59  E1 000CD        BBC     PRV, @RET_NUMBER, 12$                        ; 0947
               63   01   A7            58  28 000D2        MOVC3   R8, 1(PRV_NAME), (STRING_PTR)                ; 0951
                         83            2C  90 000D7        MOVB    #44, (STRING_PTR)+                           ; 0952
                         57   01 A847      9E 000DA 12$:   MOVAB   1(R8)[PRV_NAME], PRV_NAME                    ; 0954
                                       59  D6 000DF        INCL    PRV                                          ; 0942
                         3F            59  D1 000E1        CMPL    PRV, #63
                                       E2  1B 000E4        BLEQU   11$
                         56            53  D1 000E6 13$:   CMPL    STRING_PTR, R6                               ; 0956
                                       02  13 000E9        BEQL    14$
                                       53  D7 000EB        DECL    STRING_PTR                                   ; 0958
          0C   BC        53            56  A3 000ED 14$:   SUBW3   R6, STRING_PTR, @RET_LENGTH                  ; 0959
                                           04 000F2        RET                                                  ; 0866
                    F8   AD            03  B0 000F3 15$:   MOVW    #3, CTRSTR_DESCR                             ; 0964
                    FC   AD     01FA   CA  9E 000F7        MOVAB   P.AAJ, CTRSTR_DESCR+4                        ; 0965
                                     FF7D  31 000FD        BRW     6$                                           ; 0966
                         0E       10   AC  D1 00100 16$:   CMPL    RET_TYPE, #14                                ; 0985
                                       07  12 00104        BNEQ    17$
                         55     01FD   CA  9E 00106        MOVAB   P.AAK, PROT_CHARS                            ; 0987
                                       05  11 0010B        BRB     18$
                         55     0201   CA  9E 0010D 17$:   MOVAB   P.AAL, PROT_CHARS                            ; 0989
                         54       C8   AD  9E 00112 18$:   MOVAB   PSTRING, PSTRING_PTR                         ; 0991
                                       50  D4 00116        CLRL    I                                            ; 0999
                    E0 AD40            54  D0 00118 19$:   MOVL    PSTRING_PTR, PRMLST[I]                       ; 0996
                         51            54  D0 0011D        MOVL    PSTRING_PTR, THIS_STRING                     ; 0997
                                       84  94 00120        CLRB    (PSTRING_PTR)+                               ; 0998
                         53       50   02  78 00122        ASHL    #2, I, R3                                    ; 0999
     52        08        BC       04   53  EF 00126        EXTZV   R3, #4, @RET_NUMBER, R2
                         57       52   0F  8D 0012C        XORB3   #15, R2, PROT_FIELD
                                       15  13 00130        BEQL    22$                                          ; 1000
                         84            3D  90 00132        MOVB    #61, (PSTRING_PTR)+                          ; 1003
                                       61  96 00135        INCB    (THIS_STRING)                                ; 1004
                                       52  D4 00137        CLRL    J                                            ; 1005
                    06   57            52  E1 00139 20$:   BBC     J, PROT_FIELD, 21$                           ; 1007
                         84          6245  90 0013D        MOVB    (J)[PROT_CHARS], (PSTRING_PTR)+              ; 1010
                                       61  96 00141        INCB    (THIS_STRING)                                ; 1011
                    F2   52            03  F3 00143 21$:   AOBLEQ  #3, J, 20$                                   ; 1005
                    CD   50            03  F3 00147 22$:   AOBLEQ  #3, I, 19$                                   ; 0992
                    F8   AD            24  B0 0014B        MOVW    #36, CTRSTR_DESCR                            ; 1016
                    FC   AD     0205   CA  9E 0014F        MOVAB   P.AAM, CTRSTR_DESCR+4                        ; 1019
                                  E0   AD  9F 00155 23$:   PUSHAB  PRMLST                                       ; 1023
                                  F0   AD  9F 00158        PUSHAB  OUTSTR_DESCR
                                  0C   AC  DD 0015B        PUSHL   RET_LENGTH
                                  F8   AD  9F 0015E        PUSHAB  CTRSTR_DESCR
               00000000G   00          04  FB 00161        CALLS   #4, SYS$FAOL
                                           04 00168        RET                                                  ; 0866
                         06       08   BC  D1 00169 24$:   CMPL    @RET_NUMBER, #6                              ; 1031
                                       05  1B 0016D        BLEQU   25$
                         51            6A  9E 0016F        MOVAB   ACP_TYPES, ACPTYP_PTR                        ; 1033
                                       21  11 00172        BRB     28$
                    50   08   BC       01  78 00174 25$:   ASHL    #1, @RET_NUMBER, R0                          ; 1035
```

```
                          51          6A40  DE 00179          MOVAL   ACP_TYPES[R0], ACPTYP_PTR                       ;
                                        16  11 0017D          BRB     28$                                             ; 1036
                          0E      08    BC  D1 0017F 26$:     CMPL    @RET_NUMBER, #14                                ; 1044
                                        06  1B 00183          BLEQU   27$                                             ;
                          51      38    AA  9E 00185          MOVAB   STATES, STATE_PTR                               ; 1046
                                        0A  11 00189          BRB     28$                                             ;
              50          08    BC      01  78 0018B 27$:     ASHL    #1, @RET_NUMBER, R0                             ; 1048
                          51      38  AA40  DE 00190          MOVAL   STATES[R0], STATE_PTR                           ;
                    0C    BC            61  9B 00195 28$:     MOVZBW  (STATE_PTR), @RET_LENGTH                        ; 1049
              66    01    A1      0C    BC  28 00199          MOVC3   @RET_LENGTH, 1(STATE_PTR), (R6)                 ; 1050
                                            04 0019F          RET                                                     ; 0866
                          50    00B0    CA  9E 001A0 29$:     MOVAB   MODES, MODE_PTR                                 ; 1057
                          52            01  D0 001A5          MOVL    #1, I                                           ; 1058
                                        15  11 001A8          BRB     32$                                             ;
                                        60  95 001AA 30$:     TSTB    (MODE_PTR)                                      ; 1060
                                        07  12 001AC          BNEQ    31$                                             ;
                          50    00B0    CA  9E 001AE          MOVAB   MODES, MODE_PTR                                 ; 1063
                                        10  11 001B3          BRB     33$                                             ; 1062
                          51            60  9A 001B5 31$:     MOVZBL  (MODE_PTR), R1                                  ; 1066
                          50       01 A140  9E 001B8          MOVAB   1(R1)[MODE_PTR], MODE_PTR                       ;
                                        52  D6 001BD          INCL    I                                               ; 1058
                    08    BC            52  D1 001BF 32$:     CMPL    I, @RET_NUMBER                                  ;
                                        E5  1B 001C3          BLEQU   30$                                             ;
                    0C    BC            60  9B 001C5 33$:     MOVZBW  (MODE_PTR), @RET_LENGTH                         ; 1068
              66    01    A0      0C    BC  28 001C9          MOVC3   @RET_LENGTH, 1(MODE_PTR), (R6)                  ; 1069
                                            04 001CF          RET                                                     ; 0866
                          50      0C    BC  3C 001D0 34$:     MOVZWL  @RET_LENGTH, I                                  ; 1074
                                        0B  11 001D4          BRB     37$                                             ;
                          20          6046  91 001D6 35$:     CMPB    (I)[R6], #32                                    ; 1075
                                        03  12 001DA          BNEQ    36$                                             ;
                                  0C    BC  B7 001DC          DECW    @RET_LENGTH                                     ; 1077
                                        50  D7 001DF 36$:     DECL    I                                               ; 1075
                                        F3  12 001E1 37$:     BNEQ    35$                                             ;
                                            04 001E3 38$:     RET                                                     ; 1096

; Routine Size:  484 bytes,    Routine Base:  _LIB$CODE + 03D9
; 1039            1097  1 END                                                  ! End of module LIB$$LEXICAL
; 1040            1098  1
; 1041            1099  0 ELUDOM




;                                  PSECT SUMMARY
;
;       Name                             Bytes                          Attributes
;
;   _LIB$CODE                                    1469  NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)




;                            Library Statistics
;
;                                                -------- Symbols --------        Pages        Processing
;       File                                     Total    Loaded   Percent        Mapped       Time
;
;   _$255$DUA28:[SYSLIB]LIB.L32;1                18619       78         0          1000          00:01.4
;   _$255$DUA28:[LIBRTL.OBJ]RTLLIB.L32;1            36       17        47             8          00:00.1




;                                  COMMAND QUALIFIERS
;
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:LIBLEXICA/OBJ=OBJ$:LIBLEXICA MSRC$:LIBLEXICA/UPDATE=(ENH$:LIBLEXICA
;       )

; Size:           916 code + 553 data bytes
; Run Time:          00:20.2
; Elapsed Time:      01:30.6
; Lines/CPU Min:      3262
; Lexemes/CPU-Min: 42347
; Memory Used:  229 pages
; Compilation Complete
```

0208 AH-BT13A-SE DIGITAL EQUIPMENT CORPORATION
VAX/VMS V4.0 CONFIDENTIAL AND PROPRIETARY